PCECT502 — ANALOG AND DIGITAL COMMUNICATION

COURSE PROJECT — STUDENT HANDBOOK

Semester S5 • B.Tech Electronics & Communication Engineering

Academic Year 2026–27 • Issued 06 August 2026 • Final submission 30 October 2026

Item	Detail
Component	Assignment / Micro-project component of CIE — 15 marks
Team size	3 students per team (4 only with prior approval)
Mode	Pure software simulation in Python. No hardware, no lab booking, no purchase required.
Abstract due	Saturday, 15 August 2026 (11:59 PM)
Progress presentation	Week of 14–19 September 2026 (slot list will be published)
Final presentation + viva	27–29 October 2026
Final submission deadline	Friday, 30 October 2026 (11:59 PM) — code, report, slides
Mapped COs	CO1, CO2, CO3, CO4 (topic-dependent) — mainly K2/K3 levels

Read this first — the one thing that matters

This project is graded on **what you understand and how hard you worked**, not on how impressive your final plot looks. A team that builds a modest simulation, hits a wall, documents honestly why it happened, and can explain every line of its code in the viva will score **higher** than a team that produces a beautiful BER curve it cannot explain. Negative results, properly analysed, are full-credit results here.

Correspondingly: a project you cannot explain is worth close to zero, no matter who or what wrote it.

1. Purpose of the Project

The lecture course gives you the equations. This project asks you to make those equations produce numbers, and then to check whether the numbers behave the way the theory says they should. That single loop — derive, simulate, compare, explain the gap — is the whole point.

Specifically, by the end of the project you should be able to:

- Take a block diagram from the syllabus (a PCM transmitter, a QPSK receiver, a zero-forcing equaliser) and turn it into working code without copying someone else's implementation.
- Produce a Monte-Carlo BER curve and place the closed-form theoretical curve on top of it — and if they disagree, find out why.
- Explain a design trade-off quantitatively (bandwidth vs. SNR, bit rate vs. quantisation noise, complexity vs. performance) instead of qualitatively.
- Judge honestly where a modern machine-learning method genuinely helps a communication receiver and where it is an expensive way to reproduce what a matched filter already does optimally.
- Write and present technical work at the standard expected in a final-year project or an internship.

2. How This Project Is Graded (Please Actually Read This)

The assessment is deliberately weighted towards process and comprehension. Three mechanisms enforce this:

2.1 The individual viva

At the final presentation, each member is questioned separately for a few minutes on **any** part of the project — including parts they did not personally write. "That was handled by my teammate" is not an acceptable answer. Divide the labour; do not divide the understanding.

2.2 The logbook

Every team maintains a plain-text file, **LOG.md**, in its repository. One short entry per working session:

```
## 2026-09-02  (Anjali, Rahul)  ~2.5 h
- Tried to match simulated BPSK BER to Q(sqrt(2Eb/N0)). Simulation was
  consistently ~3 dB better than theory.
- Suspected noise variance. We had used sigma = sqrt(N0) instead of
  sqrt(N0/2) per dimension. Fixed -> curves now overlap within 0.1 dB.
- Open question: why does the overlap break down below Eb/N0 = 0 dB?
  (too few error events? need more symbols) -> to check next session.
```

The logbook is graded on **specificity and honesty, not on success**. Entries recording failed attempts, wrong assumptions and dead ends are worth more than entries saying "worked on code". A logbook with three entries all dated the last week of October is a strong signal, and it will be read as one.

2.3 Analysis questions outweigh results

Every project must answer a short set of "why" questions in the report (see the sample project in Section 10 for what these look like). Marks for these exceed the marks for the results themselves.

This scores well	This scores poorly
A simple simulation whose every parameter you can justify	An elaborate simulation with unexplained magic constants
"Our neural equaliser lost to MMSE by 1.2 dB, and here is our explanation of why"	"Our neural equaliser achieved 99.7% accuracy" (accuracy of what? at what SNR?)
Theory curve plotted on top of the simulated curve	Simulated curve alone, axes unlabelled, no reference
Weekly logbook with dead ends recorded	Logbook written the night before submission
Honest statement: "we could not get the PLL to lock; here is what we tried"	Silence about the part that did not work

3. Scope, Tools and Constraints

3.1 What you must do

- **Simulate in Python.** The core signal-processing chain must be your own code built on NumPy/SciPy. You may not hand the whole system to a library that does communication blocks for you.
- **Cover at least one full syllabus module in depth,** and connect it to at least one other.
- **Produce at least one quantitative comparison against theory** — a BER curve against the erfc/Q expression, an SQNR measurement against 6.02n + 1.76 dB, a Carson's-rule bandwidth check, a PSD against its analytical form. This is compulsory for every topic.
- **Make it reproducible.** Fixed random seeds, a `requirements.txt`, and a single command that regenerates every figure in the report.

3.2 What you must not do

- No hardware, no SDR dongles, no lab equipment. Everything runs on a laptop.
- No submitting a downloaded GitHub project with the variable names changed. We check, and it is treated as plagiarism under institute policy.
- No black-box toolbox calls for the block you are supposed to be studying. If your project is about equalisation, you write the equaliser; you may use `scipy.signal.lfilter` to run it.

3.3 Allowed software

Category	Packages
Core (expected)	Python 3.10+, NumPy, SciPy, Matplotlib
Audio / data handling	soundfile or scipy.io.wavfile, pandas, librosa (feature extraction only)
Machine learning	scikit-learn, PyTorch or TensorFlow/Keras (CPU is sufficient for everything suggested here)
Optional convenience	scikit-commpy (for cross-checking your own blocks only, never as a substitute), Streamlit or ipywidgets for a dashboard, Jupyter for exploration

Category	Packages
Free compute	Google Colab free tier is more than adequate. No paid service is required for any suggested topic.

Datasets, if your topic needs one: RadioML 2016.10a (modulation classification), any public speech corpus or your own recorded WAV files, LibriSpeech samples. All are free downloads. Cite whatever you use.

4. Policy on AI Tools and Academic Honesty

You may use ChatGPT, Claude, Copilot or similar tools while doing this project. They are part of how engineering is done now, and pretending otherwise helps nobody. The conditions are simple and they are strictly enforced:

1. **Declare it.** The report contains a short appendix titled "Use of AI tools" listing which tools you used and for what — debugging, plotting, drafting text, explaining a concept, generating boilerplate. A truthful declaration costs you no marks. An undeclared one that surfaces in the viva costs you the component.
2. **Own every line.** In the viva you may be pointed at any line of your code and asked why it is there, what happens if a parameter is doubled, and what breaks if it is removed. If you cannot answer, that code is treated as not yours.
3. **Never trust it on the physics.** Language models are confidently wrong about factor-of-two conventions in noise variance, about single-sided versus double-sided PSD, and about erfc versus Q. Verify everything against Haykin or Lathi. Several past projects have been sunk by an AI-suggested `sigma = sqrt(N0)`.
4. **Do not fabricate.** Inventing results, hand-drawing a plot that no code produced, or reporting numbers you did not measure is academic misconduct and is handled as such.

> **Note the distinction**
>
> Using AI **as a tool to build** your project is permitted and declared. Using AI **as the subject** of your project — a neural equaliser, a modulation classifier — is a technical choice and is entirely separate from the honesty policy. Many good projects do both; several excellent projects use no machine learning at all.

5. Deliverables at a Glance

#	Deliverable	Contents	Due
D1	Abstract (1 page PDF)	Title, team, problem statement, system to be built, method, expected results, tool list, references	17 Aug 2026
D2	Progress presentation	8–10 slides + live run of whatever works so far. All members present.	14–19 Sep 2026
D3	Code repository	Source, README, requirements.txt, LOG.md, figures/ folder, seeds fixed	30 Oct 2026
D4	Final report (PDF)	10–15 pages, IEEE two-column or the department template	30 Oct 2026
D5	Final presentation + individual viva	12 min talk + 5 min demo + 8 min questions	27–29 Oct 2026

6. Timeline and Milestones

Twelve working weeks, running alongside your regular course load. The plan assumes roughly **3–4 hours per person per week** — about 40 hours per person total. It is scoped so that this is genuinely enough, provided the work is spread out. It is not enough if compressed into October.

Week	Dates	What you should be doing	Milestone
W0	06 – 17 Aug	Form teams. Read the topic list. Read one textbook chapter and one paper on your shortlisted topic. Discuss scope with the faculty in charge. Write the abstract.	D1 Abstract — 17 Aug
W1	17 – 22 Aug	Abstract feedback returned by 20 Aug. Adjust scope if asked. Set up repository, environment, LOG.md. Reproduce one textbook figure — any one — to prove the toolchain works.	Repo live
W2	24 – 29 Aug	Build the signal source and transmitter blocks. Verify in time and frequency domain. First entries in the logbook.	Tx working
W3	31 Aug – 05 Sep	Build the channel model (AWGN first, always). Confirm your noise variance convention by an independent measurement of the generated noise power.	Channel verified
W4	07 – 12 Sep	Build the receiver. Get an end-to-end BER (or SQNR, or output-SNR) number out of the system. Compare against theory.	End-to-end baseline
W5	14 – 19 Sep	Prepare and deliver the progress presentation.	D2 Progress talk

Week	Dates	What you should be doing	Milestone
W6	21 – 26 Sep	Act on progress-review feedback. Begin the advanced / comparison stage (second technique, ML block, impairment study).	—
W7	28 Sep – 03 Oct	Advanced stage continues. This is the week things break; leave room for it.	—
W8	05 – 10 Oct	Complete the comparison. Generate the full result set across the SNR / bit-rate / roll-off sweep.	Results complete
W9	12 – 17 Oct	Analysis week. Answer the "why" questions. Re-run anything that looks suspicious. Finalise all figures with proper axes, units and legends.	Figures frozen
W10	19 – 24 Oct	Write the report. Code freeze on 24 Oct — after this, only documentation changes.	Draft report
W11	26 – 30 Oct	Final presentation and viva. Submit report, slides, repository.	D4/D5 — 30 Oct

Slack is built in, and it is real

W7 exists because something always fails in W7. If you are on schedule at the end of W6, you are in good shape. If you have not produced an end-to-end baseline result by the progress presentation in W5, you are behind and should come and talk to us that week rather than in October.

7. Deliverable Specifications

7.1 D1 — Abstract (due 17 August)

One page, PDF, single column, 11 pt. Filename: `S5_ADC_Abstract_<TeamNumber>.pdf`. Structure:

- **Title** — specific. "Digital Communication Using Python" is not a title. "Comparison of Zero-Forcing, MMSE and Neural-Network Equalisation on a Three-Tap ISI Channel" is.
- **Team** — names, roll numbers, and a one-line statement of who intends to own which part.
- **Problem statement** (4–6 lines) — what specific question does the project answer?
- **System to be simulated** (6–8 lines) — the block chain, in words. Name the modulation, the channel model, the receiver structure.
- **Methodology** (4–6 lines) — what you will implement, what you will sweep, what you will measure.
- **Expected results** (3–4 lines) — name the figures you expect to produce and what you expect them to show. It is fine to be wrong later; state a prediction now.
- **Validation against theory** (2 lines) — which closed-form expression will you check your simulation against?
- **Tools and references** — packages, and at least three references, one of which must be a textbook from the syllabus.

Each team should Submit a hardcopy of the abstract to me by 15/08/2026 and also mail the softcopy to ameenudeen@cet.ac.in

7.2 D2 — Progress Presentation (14–19 September)

Fifteen minutes total: about 8 minutes of slides, 3 minutes of running code, 4 minutes of questions. Every member speaks. Suggested slide set:

1. Title and team; the question you are answering in one sentence.
2. Block diagram of the system, with the blocks you have already built marked in a different colour.
3. The theory you are relying on — the key equation, stated correctly.
4. Implementation status: what runs today.
5. Two or three results so far, however preliminary.
6. **The problem slide** — what is currently broken or confusing. This slide is compulsory and it is where most of the useful feedback comes from. Teams that claim everything is fine are asked harder questions, not easier ones.
7. What remains, mapped onto the weeks W6–W10.
8. An extract from your logbook showing a real debugging episode.

7.3 D3 — Code Repository

A GitHub/GitLab repository (private, with faculty added) or a zip archive. Required layout:

```
team07_adc_project/
├── README.md            # what it is, how to run it, one-line result summary
├── LOG.md               # the working logbook
├── requirements.txt
├── run_all.py           # regenerates EVERY figure in the report
├── src/
│   ├── transmitter.py
│   ├── channel.py
│   ├── receiver.py
│   └── equalizer.py
├── tests/               # sanity checks (see below)
├── figures/             # PNG/PDF outputs, named as in the report
└── notebooks/           # exploratory work (optional, may be messy)
```

Sanity tests are compulsory — at minimum three, and they are easy marks:

- With the channel disabled and no noise, BER must be exactly 0.
- The measured variance of your generated noise must match the value you intended, to within a percent.
- Transmitting a known short sequence and receiving it back must return the identical bits.

Fix your seeds. `rng = np.random.default_rng(2026)`. A result that changes every run is a result nobody can check.

7.4 D4 — Final Report

10–15 pages including figures, IEEE two-column format (or the department's project template). Expected sections:

Section	Content	Approx. length
Abstract	150–200 words: problem, method, headline result, one number	0.2 page
1. Introduction	Motivation, what question the project answers, contribution of each stage	1 page
2. Theory	The equations you actually use, derived or cited, with your notation defined. Do not reproduce a textbook chapter.	2–3 pages
3. System model & implementation	Block diagram, parameter table (sampling rate, symbol rate, filter length, roll-off, number of symbols simulated), design decisions and why	2–3 pages
4. Results	Every figure captioned, axes labelled with units, theory overlaid where applicable	3–4 pages
5. Analysis & discussion	The "why" questions answered. Where theory and simulation disagree, and your explanation.	2–3 pages

Section	Content	Approx. length
6. Limitations & future work	Honest. What you would do with another month.	0.5 page
7. Conclusion	What you now know that you did not in August	0.3 page
References	IEEE style, minimum 6	0.3 page
Appendix A	Contribution statement — who did what, signed by all members	—
Appendix B	Use of AI tools declaration	—

Figure standards (marks are lost here routinely): axes labelled with quantity and unit; BER plots on a semi-log y-axis; theoretical curve as a solid line and simulation as markers; legends on every plot; the number of symbols simulated stated in the caption; no screenshot of a plot pasted from a phone camera.

7.5 D5 — Final Presentation and Viva (27–29 October)

25 minutes per team: 12 minutes of presentation, 5 minutes running your code live, 8 minutes of questions in which each member is addressed individually. Bring the code on a laptop that can run it offline. Sample questions from previous years:

- "Point at the line where the noise is added. Why is the variance N0/2 and not N0?"
- "Your BER at 8 dB is 10^{-4}. How many symbols did you simulate, and is that enough to trust that number?"
- "Double the roll-off factor. What happens to the bandwidth, and what happens to the eye opening?"
- "Your equaliser has 11 taps. Why 11? What happens with 5?"
- "Your neural network beats MMSE here. Is that possible in principle? What is it exploiting?"

8. Evaluation Rubric (15 marks)

Criterion	Marks	What earns full marks
Individual understanding (viva)	4	Every member can explain the theory, the code and the results, including parts they did not write. Answers show they know why choices were made, not just what was done.
Effort and process (logbook, incremental progress, response to feedback)	3	Regular dated entries across all 12 weeks, failures recorded, progress-review feedback visibly acted upon.
Progress presentation	2	Honest status, working baseline demonstrated, a real problem identified and discussed.
Analysis quality	2	Simulation compared against theory; discrepancies investigated rather than ignored; the "why" questions answered with reasoning.
Report and figures	2	Clear structure, correct notation, publication-standard figures, complete parameter table, honest limitations section.
Final presentation and demo	2	Code runs live, timing respected, all members contribute, questions handled.
TOTAL	15	

Note what is **not** in the table: there are no marks for how good your results are. There are no marks for using machine learning. There are no marks for volume of code.

Penalties

−3 No logbook, or a logbook clearly written retrospectively.

−2 Code that does not run on the evaluator's machine, or figures that cannot be regenerated.

−2 Missing or false AI-use declaration.

Zero, and referral for plagiarised code or fabricated results.

Late submission: 2 marks per day, up to three days; no submission accepted after 2 November.

9. Working as a Team

A workable split for a team of three, and the one we recommend:

- **Member A — Transmitter and channel.** Source, modulator, pulse shaping, channel models, noise generation. Owns the sanity tests.
- **Member B — Receiver and detection.** Matched filtering, synchronisation, decision logic, BER counting, theoretical curve computation.
- **Member C — Advanced stage and analysis.** The equaliser / ML block / impairment study; runs the sweeps; produces the figures; owns the report structure.

Rotate the pair-programming: whoever owns a block should walk the other two through it in a 20-minute session before that block is declared done. Record those sessions in the logbook. This is the single most effective thing you can do to protect your viva marks.

If a team member stops contributing, raise it at the progress presentation, not in the final week. The contribution statement in Appendix A is signed by all members and evaluators may award differentiated marks.

10. A Full Worked Example of a Project

What follows is a complete project specification at exactly the depth and scale we expect. It is written out in full so you know what "done" looks like. **You may choose this topic** — it is on the list at S12 — but if you do, expect harder questions in the viva, since the design is handed to you. Most teams should use it as a template for scoping their own.

This is a **depth** project: one narrow question, studied hard. Section 11 gives a second worked example of the opposite kind — a **breadth** project that builds a whole communication chain end to end. Both are worth the same marks. Read both before deciding which shape suits your team.

Learning to Undo a Channel: Zero-Forcing, MMSE, LMS and Neural-Network Equalisation of an ISI Channel

10.1 The question

A bandlimited channel spreads each transmitted symbol into its neighbours. The syllabus gives you the Nyquist criterion for zero ISI and the zero-forcing equaliser as the classical answer. The zero-forcing equaliser has a well-known weakness: it inverts the channel, so wherever the channel response is small, the equaliser gain is large, and it amplifies noise. This project asks: **how much of that penalty can be recovered, and by what?** Four answers are compared — the classical ZF equaliser, the MMSE equaliser, the adaptive LMS equaliser, and a small neural network trained on a pilot sequence.

10.2 Why this is a good project

- It sits squarely in **Module 3** (ISI, Nyquist criterion, raised cosine, equalisation, ZF equaliser) and reaches into **Module 4** (BPSK, BER vs SNR, erfc).
- It maps to **CO3 (K3)** directly and **CO4** partially.
- Every stage has a closed-form result to check against, so you always know whether your code is right.
- The AI component is genuinely motivated rather than bolted on, and — importantly — it is **not guaranteed to win**, which makes the analysis interesting.
- The baseline half is achievable by mid-September even if the ML half never works.

10.3 The system

Baseband, discrete time, one sample per symbol after matched filtering (with an oversampled version for the eye diagrams):

```
bits -> BPSK map -> upsample -> RRC pulse shape
                                      |
                                 ISI channel h[n]
                                      |
                                   + AWGN
                                      |
                        RRC matched filter -> symbol-rate sampler
                                      |
                              EQUALISER  (ZF | MMSE | LMS | MLP)
                                      |
                             threshold detector -> bits -> BER
```

Baseline parameters (state these in your report's parameter table and justify each one):

Parameter	Value	Why
Modulation	BPSK	Real-valued, so ISI and equalisation are visible without complex-plane clutter
Pulse shape	Root-raised cosine, roll-off $\alpha = 0.35$, span 8 symbols	Standard; α is one of your sweep variables
Oversampling	8 samples/symbol	Enough to draw a readable eye diagram
Channel	h = [0.407, 0.815, 0.407] (Proakis 'Channel B')	Textbook benchmark with a deep spectral null — this is what makes ZF fail
Second channel	h = [0.227, 0.460, 0.688, 0.460, 0.227]	Harder case, for the sweep
Noise	Real AWGN, variance $\sigma^2 = N0/2$ per dimension	Get this right first; most bugs live here
Eb/N0 range	0 to 14 dB in 1 dB steps	Covers the region where curves separate
Symbols per SNR point	≥ 100 / target BER (e.g. 10^6 for BER ≈ 10^{-4})	You need ≥ 100 error events for a trustworthy point
Equaliser length	11 taps (sweep 3 – 21)	Odd, centred; the sweep is part of the result

10.4 Stage-by-stage build plan

Weeks	Stage	What you build and what you must verify
W1–W2	Stage 0 — Transmitter	BPSK mapper, upsampler, RRC filter. Verify: the RRC convolved with itself gives a raised cosine with zeros at every symbol instant; plot it and check. Plot the spectrum and confirm the bandwidth is (1+α)/2T.
W3	Stage 1 — AWGN channel, no ISI	Add noise; matched filter; detect. Verify: simulated BER lies on top of Q(√(2Eb/N0)) across the whole range. Until this holds to within 0.2 dB, do not proceed. This checkpoint catches the σ = √(N0/2) error, the energy-normalisation error and the filter-delay alignment error, all of which are near-universal.
W4	Stage 2 — ISI channel	Insert h[n]. Observe: the eye closes; BER flattens into an error floor that no amount of SNR removes. Plot |H(f)| and locate the null. This is the motivation figure for the whole report.
W5	Progress presentation	Show Stages 0–2 working. Your 'problem slide' is whatever is still misbehaving.
W6	Stage 3 — Zero-forcing equaliser	Build the (2N+1)-tap ZF equaliser by solving the Toeplitz system C·w = δ. Verify: the combined channel-plus-equaliser impulse response has zeros at ±1, ±2 … symbol offsets, as the design demands. Then observe the catch: BER improves at high SNR but is *worse than no equaliser* at low SNR. Explain it with the |W(f)| plot.
W7	Stage 4 — MMSE and LMS	MMSE: same Toeplitz system with $\sigma^2 I$ added to the diagonal. LMS: adaptive, trained on a 200-symbol pilot, then decision-directed. Verify: MMSE → ZF as $\sigma^2 \to 0$. Plot the LMS learning curve (MSE vs iteration) for three step sizes and show one that diverges.
W8	Stage 5 — Neural equaliser	An MLP taking the same 11-sample window the linear equalisers see, with 2 hidden layers (e.g. 32 and 16 units, tanh or ReLU), one output, trained on pilot symbols at each SNR. Verify: with the ISI removed, the network must reduce to a threshold detector — check that it does.
W9	Stage 6 — The sweeps	BER vs Eb/N0 for all four equalisers, both channels. BER vs number of taps. BER vs pilot length for LMS and MLP. Complexity table in multiply-accumulates per symbol.
W10	Stage 7 — Analysis and writing	Answer the questions in 10.6.

10.5 Required figures

1. RRC and RC pulse in time; RC zero-crossings at symbol instants marked.
2. Transmit spectrum for α = 0, 0.35, 1.0, with the (1+α)/2T bandwidth marked.
3. Eye diagram: clean AWGN channel vs. ISI channel, same SNR, side by side.
4. |H(f)| of both ISI channels, with the spectral null annotated.
5. BER vs Eb/N0 for AWGN only: simulation markers on the theoretical Q-function line.

6. BER vs Eb/N0: no equaliser (showing the floor), ZF, MMSE, LMS, MLP, all on one semilog plot, with the AWGN bound as the reference.
7. |W(f)| of the ZF equaliser overlaid on |H(f)|, showing gain peaking at the null — the noise-enhancement figure.
8. LMS learning curves for three step sizes, including a divergent one.
9. MLP training and validation loss; BER vs pilot-sequence length.
10. BER vs equaliser length at a fixed Eb/N0 of 10 dB.

10.6 The analysis questions you must answer

These carry more marks than the figures. Answer them in Section 5 of your report, with reference to your own data.

1. Why does the ISI channel produce a BER **floor**, rather than just a shifted curve? What is the floor's value determined by?
2. The ZF equaliser is designed to eliminate ISI completely, and it does. Why, then, does it perform *worse* than doing nothing at 2 dB? Answer using your |W(f)| plot and the noise-enhancement factor.
3. Write down the MMSE tap equation and identify, in it, the exact term that fixes the ZF problem. What does that term do at high SNR, and does your simulation confirm it?
4. Your LMS equaliser converges to (approximately) the MMSE solution. What determines how fast, and what determines the excess MSE it settles at? Support with your step-size sweep.
5. Does the MLP beat MMSE? Whichever answer you get: is it possible in principle for it to do so? (Hint: think about what the *optimal* detector for this channel is — a linear filter is not it. Look up maximum-likelihood sequence detection and comment, even without implementing it.)
6. How many multiply-accumulates per symbol does each equaliser need? At what performance gain does the MLP's cost become defensible?
7. Your BER estimate at 10^{-5} is based on a finite number of trials. Give the approximate confidence interval on that point and state how many symbols you would need for a trustworthy 10^{-6}.
8. If you were allowed to change the transmitter, would you rather have a better equaliser or a different pulse shape? Justify.

10.7 Realistic effort estimate

For a team of three: about 12 person-hours on Stages 0–1, 8 on Stage 2, 15 on Stages 3–4, 20 on Stage 5, 15 on the sweeps, 25 on analysis, writing and slides. Roughly **95–115 person-hours** across twelve weeks, which is close to 3.5 hours per person per week. The Stage 5 estimate is the one that expands if you have never used PyTorch; if that is the case, do Stage 5 with `sklearn.neural_network.MLPRegressor` instead — it is completely acceptable here and it is four lines of code.

10.8 If you finish early

- Extend to QPSK and re-run everything on the complex plane.
- Add a decision-feedback equaliser and compare it with the linear ones.
- Implement the Viterbi/MLSE detector and show where the true optimum lies relative to all four of your curves. This is the most interesting extension.
- Make the channel time-varying and compare LMS tracking against retraining the network.

11. A Second Worked Example — A Breadth Project

The project in Section 10 goes deep into one block. This one goes wide: it takes a real signal — your own recorded voice — and carries it through every stage the syllabus describes, from the microphone to a constellation diagram and back to audible speech. It touches all four modules. It is the topic listed at **S21**, specified here in full.

The trade-off is real and you should understand it before choosing. A breadth project has more blocks, so more can break, and the analysis for any single block is necessarily shallower. What it gives you in return is the thing that is hardest to get from a lecture course: a felt sense of how the pieces connect, and in particular of **what a bit error actually sounds like**.

From Microphone to Constellation: An End-to-End Speech Transmission Link

11.1 The question

Speech is an analog signal. To send it over a digital radio link you must sample it, quantise it, turn it into bits, modulate a carrier, survive a noisy channel, and undo all of that at the far end. Every one of those steps loses something. The project asks: **where does the quality actually go, and which impairment dominates?** Concretely — at a fixed bit rate and a fixed transmit power, is your reconstructed voice limited by the quantiser, or by the channel? And at what Eb/N_o does the answer change?

The payoff is a single result that most students find genuinely surprising: the relationship between bit error rate and perceived audio quality is nothing like linear. A BER of 10^{-5} is inaudible; 10^{-3} is unpleasant; 10^{-2} is unintelligible. You will be able to hear all three.

11.2 Why this is a good project

- Covers **Module 1** (spectra, noise), **Module 2** (sampling, anti-aliasing, quantisation, SQNR, companding, PCM) and **Module 4** (BPSK/QPSK/QAM, constellations, BER vs erfc), with an optional reach into **Module 3** (pulse shaping, matched filtering, ISI).
- Maps to **CO1, CO2 and CO4**; CO3 if you add the equaliser extension.
- Three independent closed-form checks are available — the sampling theorem, SQNR = 6.02n + 1.76 dB, and the erfc BER expressions — so you can validate each third of the chain separately.
- It fails gracefully. If the 16-QAM stage never works, you still have a complete working link at QPSK.
- It produces an artefact you can play out loud in the viva, which makes the demo genuinely interesting to sit through.
- **No hardware.** A laptop microphone or a phone voice memo is the entire acquisition system. A pre-recorded WAV will be provided if you would rather not record your own.

11.3 The chain

```
[1] record voice (44.1 kHz, 16-bit)
      |
[2] spectral analysis  ->  choose target rate fs
      |
[3] anti-aliasing LPF  ->  [4] decimate to fs
      |
[5] companding (mu-law)  ->  [6] uniform quantiser, n bits
      |
[7] PCM bitstream  ->  [8] Gray mapping to symbols
      |
[9] RRC pulse shaping  ->  [10] carrier modulation (BPSK/QPSK/16-QAM)
```

```
      |
[11] + AWGN   <-- Eb/N0 sweep
      |
[12] matched filter -> [13] sampler -> [14] detector
      |                                    |
      +--> constellation plot          BER counter
      |
[15] bits -> samples -> expander -> [16] interpolate to 44.1 kHz
      |
[17] play, and measure output SNR / segmental SNR
```

11.4 Baseline parameters

Parameter	Value	Why this value
Recording	Mono, 44.1 kHz, 16-bit, 8–15 s of connected speech	Long enough to hear degradation; short enough that a full sweep runs in under a minute
Target sampling rate	8 kHz	Telephone standard. You must *derive* this from your own recording's spectrum, not assume it
Anti-aliasing filter	Order-8 Butterworth or 101-tap FIR, cut-off 3.4 kHz	Filter order and cut-off are both sweep variables
Companding	μ-law, μ = 255	Compare against A-law and against no companding at all
Quantiser	8 bits (sweep 3–12)	8-bit μ-law = 64 kbit/s, the G.711 rate
Bit rate	64 kbit/s	Fixed across all modulation schemes so the comparison is fair
Modulations	BPSK, QPSK, 16-QAM	Symbol rates 64, 32 and 16 kBd respectively at the same bit rate
Pulse shaping	RRC, roll-off α = 0.35, span 8, 8 samples/symbol	Gives transmit bandwidth (1+α)·Rs
Channel	AWGN, real or complex baseband equivalent	Complex baseband for QPSK/QAM; state the convention explicitly
Eb/N_o sweep	0 to 16 dB in 1 dB steps, plus audio rendered at 4, 7, 10 and 14 dB	The audio points are the ones you play in the viva
Synchronisation	Genie-aided initially (known delay); preamble correlation as an extension	Do not let sync eat your whole October

11.5 Stage-by-stage build plan

Weeks	Stage	What you build and what you must verify
W1	Stage 0 — Acquisition and analysis	Record your voice. Load it, plot the waveform and the spectrogram, and estimate the occupied bandwidth. Verify: compute the crest factor (peak-to-RMS ratio) of your own speech and report it. This number is the entire justification for companding later, so get it early. Deliverable: a one-paragraph

Weeks	Stage	What you build and what you must verify
		argument for why 8 kHz is a defensible target rate for this recording, and what you lose by choosing it.
W2	Stage 1 — Anti-aliasing and sampling	Decimate to 8 kHz, twice: once *with* the anti-aliasing filter and once deliberately *without* it. Verify: the aliased version's spectrum shows high-frequency energy folded back into the audible band; the two files sound audibly different and you can point at the fold in the spectrogram. This deliberate mistake is one of the best figures in the report — do not skip it.
W3	Stage 2 — Quantisation and SQNR	Uniform quantiser, n = 3…12 bits. Measure SQNR. Verify: plot measured SQNR against the theoretical 6.02n + 1.76 dB line. For speech it will sit *below* the line — explain why, using your crest factor from W1. Then add μ-law companding and show the gap close.
W4	Stage 3 — PCM bitstream and mapping	Serialise to bits (state your bit ordering and stick to it). Gray-map to BPSK and QPSK symbols. Verify: map and unmap with no channel — the recovered WAV must be bit-identical to the quantised input. Until this passes, everything downstream is untrustworthy.
W5	Progress presentation	Show a working end-to-end noiseless link: voice in, same voice out, through PCM and BPSK. This is a realistic and sufficient W5 target.
W6	Stage 4 — Pulse shaping and the channel	RRC filtering, upconversion to a baseband-equivalent carrier, AWGN at a specified Eb/N_o. Verify: measure the noise variance you actually generated and confirm it matches the Eb/N_o you asked for. Plot the transmit spectrum and confirm bandwidth $(1+\alpha)\cdot Rs$.
W7	Stage 5 — Receiver and BER	Matched filter, symbol-rate sampling, decision. Verify: simulated BER for BPSK and QPSK falls on $Q(\sqrt{(2Eb/N_o)})$; 16-QAM falls on its own erfc expression. Produce constellation scatter plots at 4, 8 and 14 dB.
W8	Stage 6 — Reconstruction and listening	De-map, expand (μ-law inverse), interpolate back to 44.1 kHz, write WAV files at each SNR point. Verify: output segmental SNR measured against the original. Render the audio set.
W9	Stage 7 — The sweeps	Output audio quality vs Eb/N_o for all three modulations at equal bit rate. Quality vs quantiser bit depth at a fixed clean channel. Gray vs binary mapping, measured in output SNR rather than BER.
W10	Stage 8 — Analysis and writing	Answer the questions in 11.7.

11.6 Required figures and artefacts

1. Waveform and spectrogram of your recording, with the occupied bandwidth marked.
2. Spectrum after correct anti-aliased decimation vs. after deliberate aliasing, side by side, with the folded components identified.
3. Measured SQNR vs. bit depth, with the 6.02n + 1.76 dB line overlaid, for uniform and for μ-law quantisation. Three curves, one plot.

4. The μ-law compression characteristic, plotted, with your speech amplitude histogram underneath it.
5. Transmit spectrum before and after RRC shaping, with (1+α)·Rs marked.
6. Constellation diagrams: QPSK at 4, 8 and 14 dB Eb/N_o, and 16-QAM at the same three points. Six panels.
7. BER vs Eb/N_o for BPSK, QPSK and 16-QAM, simulation markers on theoretical lines, all on one semilog axis.
8. **The headline figure:** output audio segmental SNR vs Eb/N_o for the three modulations at equal bit rate, with the BER axis on top. This is where the cliff effect becomes visible.
9. Bit-position sensitivity: output SNR degradation caused by forcing errors in the MSB only vs the LSB only, at equal BER.
10. A parameter table and a bandwidth-vs-power summary: what each modulation costs in Hz and what it costs in dB.

Artefacts to bring to the viva: the original recording, and reconstructions at Eb/N_o = 4, 7, 10 and 14 dB for QPSK, plus one 16-QAM file at 10 dB. Have them ready to play without an internet connection.

11.7 The analysis questions you must answer

1. You chose 8 kHz. Using your own recording's spectrum, quantify what that choice discards, and say whether it is audible. Would 6 kHz have been acceptable? Would 16 kHz have been worth the doubled bit rate?
2. Your measured uniform-quantiser SQNR is below 6.02n + 1.76 dB. By how much, and why? Connect the answer numerically to the crest factor you measured in Week 1.
3. μ-law companding improves SQNR for speech but makes it *worse* for a full-scale sine wave. Demonstrate both, and explain the mechanism in one paragraph.
4. At a fixed 64 kbit/s, 16-QAM occupies a quarter of the bandwidth of BPSK. What does it cost, in dB of Eb/N_o, for the same BER? Read the number off your own curves and compare it against the theoretical prediction.
5. Explain the shape of your headline figure. Why is output audio quality almost flat over a wide range of Eb/N_o and then collapse over about 3 dB? What sets the location of that cliff?
6. An error in the most significant bit of a sample does far more damage than an error in the least significant bit, yet the BER counter treats them identically. Quantify the difference from your Figure 9, and say what a system designer should do about it.
7. Does Gray mapping help the *audio* as much as it helps the *BER*? Measure both and explain any difference.
8. You transmitted a 64 kbit/s stream. What is the minimum bandwidth required by the Nyquist criterion for each modulation, what did you actually use with α = 0.35, and what does the Shannon capacity formula say about whether your operating point is anywhere near optimal?

11.8 Realistic effort estimate

For a team of three: roughly 10 person-hours on Stages 0–1, 12 on quantisation and SQNR, 12 on the PCM/mapping chain, 15 on pulse shaping and channel, 15 on the receiver and BER validation, 10 on reconstruction and audio rendering, and 25 on sweeps, analysis, writing and slides. About **95–105 person-hours** over twelve weeks. The risk is not any single stage — none is hard — but the number of interfaces between stages. Agree on your array shapes and bit ordering in Week 4 and write them down.

11.9 If you finish early

- Replace PCM with DPCM or ADPCM at the same bit rate and re-run the whole comparison. This turns the project into a proper Module 2 study.
- Add a Hamming or convolutional code and show the coding gain on both the BER curve and the audio.
- Add a multipath channel and an equaliser — which connects this project directly to Section 10's.
- Replace genie-aided synchronisation with a real preamble-correlation timing recovery and quantify the penalty.
- Build a Streamlit dashboard exposing bit depth, modulation and Eb/N_o as sliders with live constellation, spectrogram and audio playback. Good demo value, but do this last — it is presentation, not physics.

Scoping warning for this project

The commonest way this project goes wrong is that the team spends six weeks on audio handling and file I/O and reaches the modulator in October. The audio is the *motivation*, not the content. Stages 0–2 should be finished in three weeks. If you are still fighting WAV formats in Week 4, drop your own recording and use the provided file.

12. Suggested Topics

Pick one of these — including either of the two worked examples — or propose your own; proposals are welcome and are approved at the abstract stage. Every topic below is doable in twelve weeks by three students with no hardware. "AI" in the last column indicates a machine-learning component; topics without it are equally acceptable and equally well marked.

#	Topic	What you build	Mods
S1	Analog Modulation Workbench	AM, DSB-SC, SSB (both filter and phase-shift methods) and VSB from scratch. Compare spectra and power efficiency; implement envelope and coherent detection; measure output SNR vs input SNR and show where envelope detection breaks down (threshold effect).	M1
S2	FM Explorer and Carson's Rule	NBFM and WBFM generation; spectra against Bessel-function predictions; numerical verification of Carson's rule as β varies; pre-emphasis/de-emphasis pair and the measured SNR improvement; demodulation by differentiator-envelope and by PLL.	M1
S3	Superheterodyne Receiver Simulator	Full RF→mixer→IF→detector chain. Demonstrate image frequency by injecting a second signal at fs+2·fIF; study the effect of IF choice on selectivity vs image rejection; add double conversion and quantify the improvement.	M1
S4	Receiver Noise Chain Analyser	Thermal and shot noise generation with the correct PSD; Friis cascade formula for noise figure; Monte-Carlo verification; noise temperature; effect of stage ordering on overall NF.	M1
S5	AI Modulation Classifier	CNN on raw IQ samples classifying AM / FM / BPSK / QPSK / 16-QAM across −10 to +20 dB SNR. Confusion matrices vs SNR. Compare against a classical decision-tree classifier using cumulants and instantaneous-feature statistics — the comparison is the point. Dataset: RadioML 2016.10a or self-generated.	M1, M4, AI
S6	Sampling and Aliasing Laboratory	Sampling, aliasing, and reconstruction with practical (non-ideal) filters; aperture effect of sample-and-hold and its compensation; audible demonstration on a speech file; verification of the sampling theorem boundary.	M2
S7	Quantiser and Companding Study	Uniform vs μ-law vs A-law on speech and on a sinusoid; numerical verification of SQNR = 6.02n + 1.76 dB and of where it fails; Lloyd–Max optimal quantiser; comparison against a k-means learned quantiser trained on the actual signal statistics.	M2, AI
S8	PCM / DPCM / Delta Modulation Codec Shootout	All three implemented on real speech at matched bit rates. Predictor order sweep for DPCM. Slope overload and granular noise demonstrated and measured in DM; adaptive DM as the fix. Quality measured by SNR and a perceptual proxy.	M2
S9	Line Code Toolbox	NRZ, RZ, AMI, Manchester, HDB3, Miller. Estimated PSD compared against the analytical expression for each; DC content;	M2

#	Topic	What you build	Mods
		bandwidth; timing-recovery behaviour on long runs; BER over AWGN with a matched filter.	
S10	Neural Audio Compressor vs ADPCM	A small autoencoder compressing speech to the same bit rate as ADPCM, with a quantised bottleneck. Rate–distortion comparison. Honest discussion of complexity and latency.	M2, AI
S11	ISI and the Nyquist Criterion	Raised-cosine roll-off swept against bandwidth, eye opening and jitter sensitivity; frequency-domain verification of the Nyquist criterion by folding the spectrum; the duobinary partial-response alternative and its cost.	M3
S12	Equaliser Shootout (the worked example, §10)	ZF vs MMSE vs LMS vs neural-network equalisation on ISI channels. Fully specified in Section 10 of this handbook.	M3, AI
S13	Optimum Receiver Structures	Matched filter and correlation receiver shown to be equivalent (numerically, sample by sample); ML vs MAP detection with unequal priors and unequal symbol energies; decision regions plotted; theoretical vs simulated error probability.	M3
S14	Signal Space Visualiser	Gram–Schmidt orthogonalisation applied to arbitrary user-supplied waveform sets; automatic constellation and decision-region plotting; union bound compared against simulated symbol error rate for several 2-D and 3-D constellations.	M3
S15	Digital Modulation BER Benchmark	BPSK, QPSK, 8-PSK, 16-QAM, 64-QAM. Simulated BER against the erfc expressions. Gray vs binary mapping. Constellation diagrams under noise. The bandwidth-efficiency vs Eb/N0 plane with the Shannon limit drawn on it.	M4
S16	Carrier and Timing Recovery	Effect of carrier phase offset, frequency offset and sampling-instant error on QPSK BER; Costas loop and squaring-loop recovery; early-late gate timing recovery; quantified BER penalty for each residual impairment.	M4
S17	End-to-End Learned Communication System	A channel autoencoder that learns its own constellation and receiver jointly over an AWGN channel. Compare the learned constellation against QPSK and 16-QAM at equal rate. Show what it learns under a peak-power constraint — the result is often visually striking and genuinely explainable.	M4, AI
S18	OFDM Mini-System	QAM mapping, IFFT/FFT, cyclic prefix, multipath channel, per-subcarrier single-tap equalisation, PAPR distribution and clipping penalty. Compare against single-carrier QAM on the same channel.	M3, M4
S19	Adaptive Modulation by Reinforcement Learning	A bandit or Q-learning agent selecting modulation order per channel state to maximise throughput subject to a target BER, compared against fixed-threshold switching. The comparison against the simple threshold rule is the interesting part.	M4, AI
S20	Neural Receiver for a Nonlinear Channel	Saleh-model power-amplifier nonlinearity applied to 16-QAM; classical predistortion vs an NN demapper; EVM and BER; discussion of why the classical optimum receiver stops being optimal here.	M4, AI

#	Topic	What you build	Mods
S21	End-to-End Speech Link (worked example, §11)	Record your own voice; anti-alias, sample, quantise, compand, PCM-encode, modulate (BPSK/QPSK/16-QAM), add AWGN, receive, plot constellations and BER, and listen to the reconstruction. Fully specified in Section 11 of this handbook.	M1–M4
S22	Denoising Autoencoder for Constellation Recovery	An autoencoder that cleans a noisy/distorted 16-QAM constellation before detection, compared honestly against the matched-filter optimum — including the case where it cannot beat it, and why.	M4, AI

13. Choosing and Scoping Your Topic

Four questions to ask before you commit:

1. **Can I state the question in one sentence, with a measurable answer?** "How much does raised-cosine roll-off cost in bandwidth for a given jitter tolerance?" is a project. "Study digital modulation" is not.
2. **Is there a closed-form result I can check against?** If there isn't one anywhere in the project, you have no way of knowing your code is correct, and neither do we.
3. **Does the first third of it work on its own?** Structure the project so that if the ambitious part fails, you still have a complete, defensible result. Every topic above is built this way.
4. **Am I choosing it because it interests me or because it sounds impressive?** Twelve weeks is long enough that the second reason will not sustain you.

On scope: the commonest failure mode in this course is not laziness, it is over-ambition. A team proposes an end-to-end 5G-like system in August and delivers a half-working QAM mapper in October. **Halve your plan at the abstract stage.** If you finish early, Sections 10.8 and 11.9 show what extension looks like — and extensions are always better received than unfinished ambitions.

14. Common Pitfalls

Collected from previous cohorts. Most projects hit at least three of these.

Pitfall	How to avoid or detect it
Noise variance off by a factor of 2 or by √2	For real baseband AWGN, $\sigma^2 = N0/2$ per dimension. Symptom: your BER curve is parallel to theory but shifted by exactly 3 dB. Check it in Week 3 and never again.
Confusing Eb/N0, Es/N0 and SNR	$Es = Eb \cdot \log_2 M$. Write the conversion explicitly in your code with a comment. Symptom: the shift between your curve and theory equals $10\log_{10}(\log_2 M)$.
erfc vs Q vs erf confusion	$Q(x) = \frac{1}{2}\cdot\mathrm{erfc}(x/\sqrt{2})$. Write a one-line helper and test it against a table value before using it anywhere.
Filter delay not compensated	Pulse-shaping filters delay the signal by (span·sps)/2 samples. Symptom: BER of about 0.5 regardless of SNR. Always cross-correlate transmitted and received sequences to find the true delay.
Too few symbols for the BER claimed	You need on the order of 100 error events per point. At BER 10^{-5} that is 10^7 bits. Report the count in every caption; a lone point at 10^{-6} from 10^4 trials is meaningless.
Not normalising energy	Normalise the pulse shape to unit energy, or your Eb bookkeeping silently breaks.
ML model that has secretly memorised the test set	Keep train / validation / test strictly separate, use different random seeds for each, and always report performance on data the model never saw at that SNR.
Neural network compared against nothing	Every ML result must be plotted against the classical baseline on the same axes. An ML curve alone is not a result.

Pitfall	How to avoid or detect it
Everything left to October	The logbook makes this visible and it is explicitly penalised. Week 7 is where projects are actually decided.
Figures generated by hand, once	Write run_all.py in Week 2, not Week 10. You will regenerate every figure at least four times.

15. Frequently Asked Questions

Question	Answer
Do we have to use AI/machine learning?	No. It is one option among several. Roughly half of the strongest projects each year use none. A carefully executed classical project outscores a shallow ML one every time.
Can we use MATLAB instead of Python?	The core must be Python — it keeps the evaluation consistent and the tooling free. You may use MATLAB or Octave to cross-check a result, and you should say so if you do.
Can two teams choose the same topic?	Yes, up to two teams per topic. Your implementations and analyses will differ, and comparing them at the final presentation is useful. You may not share code.
We want to propose our own topic. Is that allowed?	Encouraged. Bring a draft abstract by 12 August so there is time to iterate before the 17 August deadline.
Our simulation disagrees with the textbook. What do we do?	First celebrate — this is the most educational thing that can happen to you. Then investigate, document the investigation in the logbook, and report the outcome. If you find the discrepancy is real and explain it correctly, you gain marks.
Do we need a GPU?	No. Every ML topic listed is trainable on a laptop CPU in minutes, or on Colab's free tier if you prefer.
How long should the code be?	There is no target. Most good submissions are 400–900 lines of source. Length is not assessed; clarity is.
What if our advanced stage completely fails?	Report it, with the evidence of what you tried. A well-documented failure with correct diagnosis scores in the upper band. Concealing it does not.
Can we reuse code from a tutorial or a paper's repository?	Small utilities, yes, with attribution in the README and a comment at the point of use. The block your project is actually about must be yours.
Is the report length strict?	10–15 pages is a guide. A dense, well-argued 11 pages beats a padded 18.

16. Final Submission Checklist — 30 October 2026

Tick every line before you submit. Items marked ▣ are the ones most often missed.

- ☐ Abstract submitted on time (15 Aug) and any requested scope change made
- ☐ Repository shared with faculty, or zip uploaded, named team<NN>_adc_project
- ☐ ▣ run_all.py regenerates every single figure in the report, from a clean checkout
- ☐ ▣ requirements.txt present, and tested in a fresh virtual environment

☐ ▣ All random seeds fixed; two consecutive runs give identical numbers

☐ Three sanity tests present in tests/ and passing

☐ ▣ LOG.md contains dated entries spanning August through October

☐ README.md states what the project does, how to run it, and the headline result

☐ Report PDF, 10–15 pages, all sections of §7.4 present

☐ ▣ Every figure has labelled axes with units, a legend, and a caption stating the number of symbols simulated

☐ ▣ At least one figure overlays theory on simulation

☐ Parameter table complete (sampling rate, symbol rate, filter span, roll-off, SNR range, trial count)

☐ Analysis section answers the 'why' questions for your topic

☐ Limitations section written honestly

☐ ▣ Appendix A — contribution statement, signed by every member

☐ ▣ Appendix B — AI tools declaration

☐ Minimum 6 references in IEEE style, including at least one syllabus textbook

☐ Final slides (max 15) uploaded

☐ Laptop tested for the live demo — runs offline, no last-minute pip install

---

Questions about scope, tools or timeline should be raised early rather than late.